Pendekodean Gabungan SPIHT-LDPC Menggunakan Teknik Diversitas

M. Agus Zainuddin dan Aries Pratiarso

Jurusan Telekomunikasi, Politeknik Elektronika Negeri Surabaya (PENS - ITS),
Surabaya 60111
E-mail: magusz@eepis-its.edu, aries@eepis-its.edu

Abstrak - Kami mengusulkan metode transmisi citra menggunakan pengkodean gabungan SPIHT-LDPC. Pada sisi pengirim, citra dikompresi menggunakan algoritma SPIHT. *Bit stream* yang dihasilkan kemudian diproses menggunakan teknik *unequal error protection* (UEP), dengan memproteksi data SPIHT yang penting menggunakan kode LDPC dengan *code rate* yang lebih rendah (kemampuan proteksi *error* yang lebih handal). Hal ini ditujukan untuk meningkatkan kinerja *bit error rate* (BER), sehingga didapatkan peningkatan nilai *peak signal to noise ratio* (PSNR). Hasil simulasi menunjukkan bahwa metode yang digunakan dapat meningkatkan performansi sistem transmisi citra.

Keyword: Pengkodean gabungan SPIHT-LDPC, Kompresi Citra SPIHT, diversity Kode LDPC, *Iterative decoding.*

1. Pendahuluan

Pada dekade terakhir ini terjadi peningkatan kebutuhan komunikasi data, baik dari segi layanan, kehandalan sistem, maupun laju transmisinya. Dengan berintegrasinya teknologi *wireless* dan layanan multimedia, pentransmisian citra dan video dengan kualitas yang baik, menjadi satu dari tujuan dari sistem jaringan *wireless* generasi selanjutnya (4G dan seterusnya).

Sinyal multimedia memiliki ukuran data yang besar, sehingga dibutuhkan teknik kompresi apabila akan ditransmisikan ataupun disimpan dalam media penyimpan data. SPIHT dan JPEG 2000 merupakan algoritma kompresi citra yang mampu mencapai rasio kompresi yang tinggi. Namun aliran data terkompresi sangat rentan terhadap gangguan kanal, meski untuk jumlah kesalahan data yang sedikit. Hal ini mensyaratkan pengkodean kanal untuk memproteksi data sebelum ditransmisikan pada kanal. Kode LDPC merupakan teknik pengkodean kanal yang mampu mendekati batas kapasitas *Shannon* [1].

Pada artikel ini kami mengusulkan sebuah metode untuk memperbaiki performansi sistem transmisi citra pada kanal *noise*. Pada sisi pengirim digunakan teknik pengkodean gabungan SPIHT-LDPC yang mengalokasikan *bit budget* secara optimal menggunakan teknik UEP. Data yang penting diproteksi menggunakan kode LDPC dengan *code rate* yang lebih rendah. Dengan demikian pada data yang penting jarang terjadi *error*, sehingga citra dapat direkonstruksi dengan kualitas yang lebih baik.

Paper diorganisasi sebagai berikut: Bagian 2 menjelaskan secara singkat tentang algoritma SPIHT dan kode LDPC. Bagian 3 membahas mengenai metode yang diusulkan. Bagian 4 membahas hasil simulasi. Serta kesimpulan pada bagian 5.

2. Kompresi Citra dan Pengkodean Kanal

2.1 Algoritma Kompresi SPIHT.

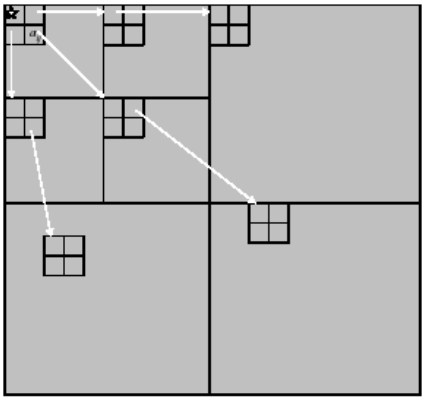
Algoritma SPIHT diusulkan oleh *Pearlman* dan *Said*, didasarkan pada transformasi *wavelet* [2]. Algoritma SPIHT secara intensif menggunakan struktur data dinamis dari koeffisien *wavelet*, untuk mengeksploitasi *self-similarities*. Hubungan *parent-child* pada koeffisien *wavelet* ditunjukkan pada Gambar 1.

Pada Algorima SPIHT koeffisien-koeffisien diklasifikasikan kedalam tiga set, yaitu:

- LIP (*list of insignificant pixel*) merupakan koordinat dari koeffisien yang tidak signifikan berdasarkan threshold saat ini.
- LSP (*list of significant pixel*) merupakan koordinat dari koeffisien yang tidak signifikan berdasarkan threshold saat ini.
- LIS (*list of insignificant sets*) merupakan koordinat dari akar dengan subpohon yang tidak signifikan.

Selama proses kompresi, set dari koeffisien pada LIS diperbaharui dan jika koeffisien menjadi signifikan dipindahkan dari LIP ke LSP. Dengan demikian *bitstream* dapat diorganisasi secara progressif.

Dengan cara yang sama set secara berurutan dievaluasi sesuai LIS, dan saat set yang ditemukan signifikan ia dihilangkan dari daftar dan dipartisi. Subset baru dengan lebih dari satu elemen ditambahkan kembali ke LIS, dengan set koordinat tunggal ditambahkan ke akhir LIP atau LSP, tergantung apakah mereka sigfikan atau tidak.



Gambar 1. Ilustrasi hubungan *parent-child* dari koeffisien SPIHT.

Algoritma pengkodean dinyatakan sebagai berikut:

1. *Inisialisasi*: Keluaran n = $[\log_2(\max_{(i,j)}\{|c_{i,j}|\}]$, set LSP dengan isi kosong, dan tambahkan koordinat $(i,j) \in$ H ke LIP, dan jika turunannya juga ke LIS dengan tipe A.
2. *Sorting pass*:
   2.1. Untuk setiap entri (i,j) pada LIP:
   2.1.1. Keluarkan $S_n(i,j)$;
   2.1.2. jika $S_n(i,j) = 1$, pindahkan (i,j) ke LSP dan keluarkan tanda dari $c_{i,j}$.
   2.2. Untuk setiap entry (i,j) pada LIS:
   2.2.1. Jika entri adalah tipe A:
   - Keluarkan $S_n(D(i,j))$.
   - Jika $S_n(D(i,j)) = 1$, maka:
   1. Untuk setiap $(k,l) \in O(i,j)$:
   - Keluarkan $S_n(k,l)$.
   - Jika $S_n(k,l) = 1$, tambahkan (k,l) ke LSP dan keluarkan tanda dari $c_{k,l}$.
   - Jika $S_n(k,l) = 0$, tambahkan (k,l) ke akhir dari LIP.
   2. Jika $L(I,j) \neq \varnothing$, pindahkan (i,j) ke akhir dari LIS, dan bila entry tipe B, menuju ke langkah 2.2.1, jika tidak hilangkan entry (i,j) dari LIS.
   2.2.2. Jika entri adalah tipe B:
   - Keluarkan $S_n(L(i,j))$.
   - Jika $S_n(L(i,j)) = 1$, maka:
     a) Tambahkan setiap $(k,l) \in O(i,j)$ ke akhir dari LIS jika entri tipe A.
     b) Hilangkan (i,j) dari LIS.
3. *Refinement Pass*: untuk setiap entry (i,j) dalam LSP, kecuali yang termasuk pada sorting pass terakhir (pada n yang sama), keluarkan bit msb ke-n dari $|c_{i,j}|$.
4. *Update* langkah kuantisasi: kurangi n dengan 1, lalu kembali ke langkah 2.

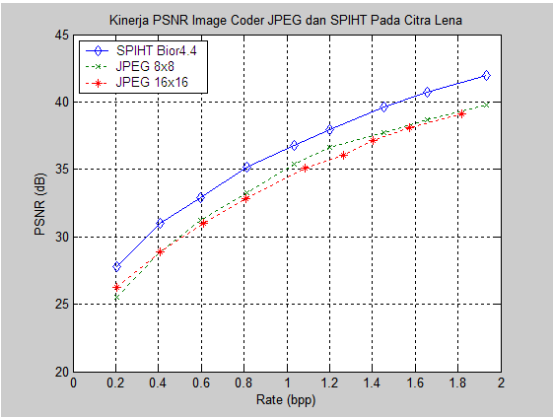
Algoritma pendekodean SPIHT menggunakan metode yang sama pengkodeannya, sehingga citra dapat direkonstruksi. Karena pada proses kompresi terjadi proses pemfilteran, maka citra hasil rekonstruksi akan mengalami distorsi. Distorsi dinyatakan dengan *mean square error* (MSE):

$$MSE = \left(\frac{1}{M.N}\right)\sum_{y=1}^{M}\sum_{x=1}^{N}\left[I(x,y) - I'(x,y)\right]^2 \qquad (1)$$

Kualitas citra hasil rekonstruksi dibandingkan dengan citra asal menggunakan parameter *peak signal to noise ratio* (PSNR). PSNR dihitung menggunakan persaman berikut:

$$PSNR = 20\log_{10}\left(\frac{2^{res}-1}{\sqrt{MSE}}\right) \qquad (2)$$

Semakin besar nilai PSNR, makan kualitas citra hasil rekonstruksi semakin baik. Hasil simulasi perbandingan kinerja algoritma kompresi SPIHT dan kompresi standar (JPEG) ditunjukkan pada Gambar 2. Dari Gambar 2 dapat ditunjukkan bahwa kompresi SPIHT memiliki nilai PSNR yang lebih tinggi dibandingkan dengan JPEG pada laju kompresi yang sama. Sehigga kinerja algoritma kompresi SPIHT lebih baik dari JPEG.



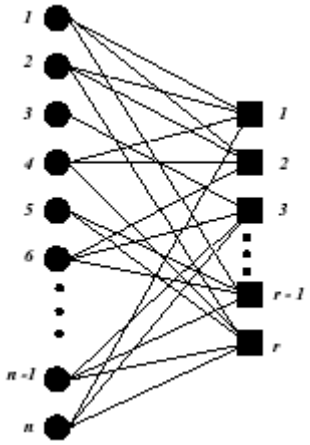
Gambar 2. Grafik perbandingan unjuk kerja algoritma kompresi SPIHT dengan JPEG.

2.2. Kode LDPC

Kode LDPC ditemukan oleh *Robert Gallager* dalam tesis PhDnya [3], dan ditemukan ulang 30 tahun sesudahnya [4],[5]. Kode LDCP merupakan kode blok linier yang diperoleh dari *sparse bipartite graph* (*Tanner graph*). Kode LDPC menggunakan matriks cek paritas yang *sparse* (jumlah elemen *non-zero* yang sedikit). Graph terdiri dari *n message* atau *bit nodes* dan *r check nodes*. *Graph* memunculkan kode blok linier dengan panjang *n*. *Codeword* merupakan vektor $(c_1,c_2,...,c_n)$ yang mana untuk seluruh *check node*, jumlah posisi bersebelahan berdasarkan *message node* adalah nol. Ilustrasi contoh dari *Tanner Graph* ditunjukkan pada Gambar 3.

Tanner graph dari kode LDPC disebut *regular* jika setiap *message node* dihubungkan dengan jumlah yang sama ke setiap *check node* dan

setiap *check node* dihubungkan dengan jumlah yang sama ke *message node*. Jika tidak kode LDPC disebut dengan *irregular*.



Gambar 3. *Tanner Graph* dari kode LDPC.

Serupa dengan kode *Turbo*, kode LDPC menggunakan pendekodean iteratif memberikan performansi BER yang sangat baik dengan kompleksitas rendah [6]. Kode LDPC *irregular* diketahui memiliki performansi yang lebih baik dari kode *Turbo* dan kode LDPC *regular*. Keuntungan dari kode LDPC *irregular* dibanding kode *Turbo* adalah proses dekoding lebih cepat, dan eror yang terjadi dapat dideteksi [7].

Kode LDPC yang digunakan untuk simulasi adalah algoritma *sum-product*. Tujuan dari pendekodean *sum-product* adalah menghitung *a posteriori probability* (APP) untuk setiap bit *codeword* $P_i = P\{c_i = 1|N\}$, yang merupakan probabilitas bit *codeword* ke-*i* adalah 1 kondisional pada kejadian N dan seluruh persyaratan cek paritas terpenuhi. Probabilitas intrinsic (probabilitas a priori) P_i^{int}, merupakan probabilitas bit awal yang tidak tergantung pada persyaratan kode, dan probabilitas extrinsic P_i^{ext} didapat dari kejadian *N*.

Algoritma *sum-product* memiliki empat langkah sebagai berikut:

1. *Inisialisasi*. Pesan inisial dikirim dari message node i ke check node j berupa LLR dari sinyal y_i (soft) yang diterima dari kanal. Untuk kanal AWGN dengan *signal-to-noise ratio* E_b/N_0:

$$L_{i,j} = R_i = 4y_i \frac{E_b}{N_0}$$

2. *Check-to-bit*: LLR extrinsic dari *check node* ke-*j* untuk *message node* ke-i merupakan probabilitas cek paritas ke-j terpenuhi jika bit ke-*i* diasumsikan 1.

$$E_{i,j} = \log_e\left(\frac{1+\prod_{i'\in B_{j,i'\neq i}} \tanh\left(L_{i',j}/2\right)}{1-\prod_{i'\in B_{j,i'\neq i}} \tanh\left(L_{i',j}/2\right)}\right) \quad (3)$$

3. *Codeword test*: Kombinasikan LLR dengan menjumlahkan LLR ekstrinsik dan LLR awal pada langkah 1.

$$L_i = \sum_{j\in A_i} E_{i,j} + R_i \quad (4)$$

Pada setiap bit lakukan *hard decision*:

$$z_i = \begin{cases} 1, & L_i \le 0 \\ 0, & L_i > 0 \end{cases} \quad (5)$$

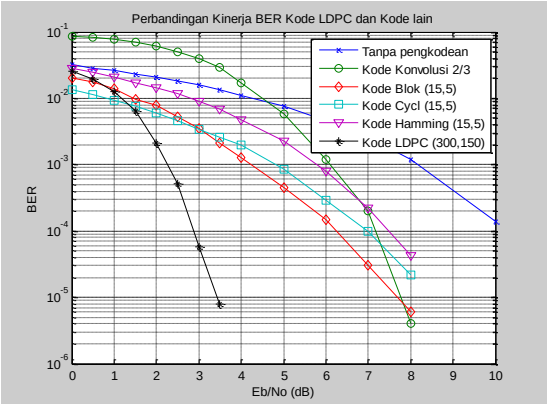
Jika z = $[z_1,...,z_n]$ merupakan *codeword* yang valid ($H.z^T = 0$), atau jika jumlah maksimum iterasi yang dijinkan terlewati, algoritma dihentikan.

4. *Bit-to-check*: Pesan dikirim dari setiap *message node ke check node* yang terhubung, kecuali *message node* ke-*i* mengirim ke *check node j*, menghitung LLR tanpa menggunakan informasi dari *check node* ke-*j*:

$$L_{i,j} = \sum_{j'\in A_{i,j'\neq j}} E_{i,j'} + R_i \quad (6)$$

Kembali ke langkah 2.

Kinerja kode LDPC dibandingkan dengan pengkodean kanal lainnya ditunjukkan pada Gambar 4.



Gambar 4. Grafik perbandingan unjuk kerja kode LDPC dengan pengkodean kanal lainnya.

3. Pengkodean Gabungan SPIHT-LDPC.

Pengkodean gabungan sumber-kanal dilakukan dengan menggunakan teknik *unequal error protection* (UEP) pada informasi yang akan ditransmisikan. Pada umumnya aliran data terkompresi memiliki sensitifitas yang berbeda terhadap *error* pada kanal. Pengkodean kanal dapat menggunakan informasi tentang sensitifitas data dari pengkode sumber. Skema pengkodean gabungan sumber-kanal ditunjukkan pada Gambar 5.

Pada sisi pengirim terdapat SSI (*Source Significance Information*) yang berfungsi memberikan informasi tentang data signifikan pada pengkode kanal dari pengkode sumber, dan RAI

(*Rate Allocation Information*) digunakan untuk mengatur *bit budget* antara pengkode sumber dan pengkode kanal. Pada sisi penerima terdapat CSI (*Channel State Information*) untuk menghitung *log-likelyhood ratio* (LLR) yang dibutuhkan oleh pendekode LDPC, DRI **(***Decoder Reliability Information*) memberikan informasi tingkat keberhasilan pendekodean data, dan BSA **(***Bit Source Allocation*) untuk mengekstraksi bit informasi dari *codeword* hasil pendekodean kanal.

Informasi yang penting diproteksi menggunakan pengkode kanal dengan *rate* yang rendah (kemampuan koreksi *error* yang handal) yaitu *rate* 1/3, sedangkan informasi yang kurang penting diproteksi menggunakan pengkode kanal dengan *rate* yang lebih tinggi yaitu *rate* 1/2. Data signifikan merupakan data yang terdapat pada awal *bit stream*. Pada simulasi dilakukan pemilihan 1/4, 1/2, dan 3/4 dari panjang awal *bit stream*.

Pada joint LDPC decoding (pendekodan gabungan LDPC), model kanal yang digunakan terdiri dari dua kanal paralel yang independen. Pada kanal ditransmisikan blok data yang sama, yang kemudian didekodekan oleh LDPC *decoder* 1 dan LDPC *decoder* 2.

Diasumsikan bahwa probabilitas kedua kanal mengalami *noise* yang besar secara bersamaan adalah kecil. Sehingga bila salah satu pendekode LDPC gagal mendekodekan data, maka kita masih dapat memperoleh data dari pendekode LDPC lainnya. Data keluaran dari *codeword selector* dipilih berdasarkan aturan:

$$\hat{x} = \begin{cases} \hat{x}_1, & \hat{c}_1.H^T = 0 \\ \hat{x}_2, & \hat{c}_1.H^T \neq 0 \end{cases}$$

Data $\hat{x}$ kemudian didekompresi menggunakan algoritma dekompresi SPIHT sehingga diperoleh citra rekonstruksi ($\hat{I}$). Kinerja BER sistem pendekodean gabungan adalah dengan membandingkan data *x* dan data $\hat{x}$, sedangkan kinerja PSNR dihitung dengan membandingkan nilai *I* dan $\hat{I}$.

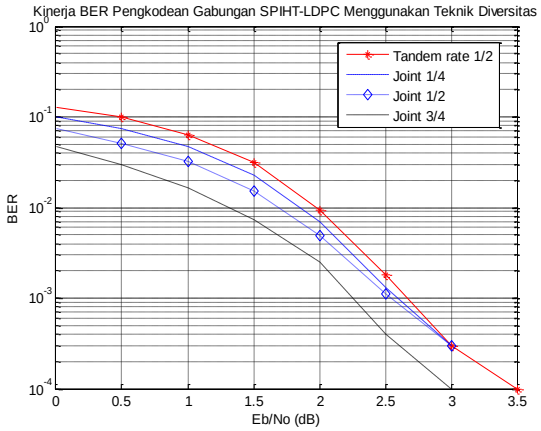
4. Hasil Simulasi

Citra yang digunakan adalah *Lena.bmp* dengan resolusi *pixel* 256x256. Citra (*I*) dikompresi menggunakan algoritma SPIHT dengan *rate* kompresi 1 *bit per pixel* (bpp), menghasilkan *bit stream* (*x*). Pada pengkodean kanal digunakan kode LDPC *irregular* dengan panjang *codeword* 300 bit. Sehingga untuk rate ½ (LDPC (300,150)), digunakan matriks cek paritas H(150,300), dan rate 1/3 (LDPC(300,100) menggunakan matriks cek paritas H(200,300). *Codeword* (*c*) yang dihasilkan kemudian dimodulasi menggunakan modulasi *binary phase shift keying* (BPSK) *baseband* (*tx*), lalu ditransmisikan melalui kanal *Additif White Gaussian Noise* (AWGN).

Sinyal yang diterima (*rx*) kemudian didemodulasi ($\hat{c}$), lalu diproses menggunkan penkode LDPC untuk koreksi *error*. Proses pendekodean iteratif menggunakan jumlah iterasi maksimum 50 kali. Keluaran dari pendekode LDPC ($\hat{x}$) kemudian diproses oleh pendekode SPIHT sehingga diperoleh citra rekonstruksi ($\hat{I}$). Nilai PSNR diperoleh dengan membandingkan citra asal dengan citra rekonstruksi.

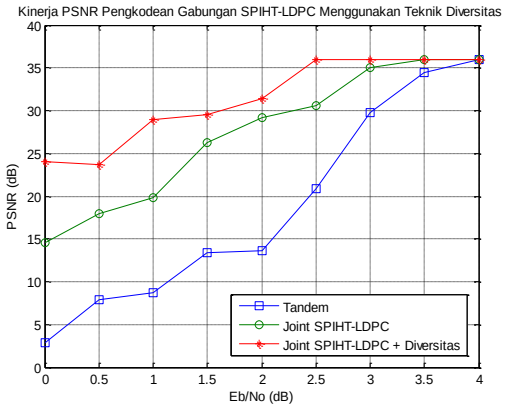
Gambar 6, menunjukkan perbandingan kinerja pengkodean gabungan dengan pengkodean tandem. pengkodean gabungan 1/4 memiliki arti 1/4 bit stream awal diproteksi dengan kode LDPC *code rate* 1/3, dan sisanya dengan *code rate* 1/2. Demikian juga untuk pengkodean gabungan 1/2 dan 3/4. Kinerja pengkodean gabungan berada diantara kinerja pengkodean tandem dengan *code rate* 1/3 dan *code rate* 1/2. Karena data terkompresi sangat rapuh terhadap error kanal, dan kesalahan bit mengakibatkan bit-bit menjadi tidak berarti. Maka proteksi bit di awal bit stream sangat penting.

Dari Gambar 7, tampak bahwa semakin banyak jumlah *bit stream* awal yang diproteksi dengan kode LDPC *code rate* 1/3, peningkatan nilai PSNR yang diperoleh cukup signifikan.



Gambar 6. Grafik perbandingan kinerja BER pengkodean gabungan 1/4, 1/2, dan 3/4, terhadap pengkodean tandem *code rate* 1/2 dan 1/3.

Dari Gambar 7, dapat dibuat tabel peningkatan nilai PSNR untuk Eb/No yang bervariasi. Pada Tabel1, tampak bahwa pada Eb/No yang rendah (2 dB) peningkatan nilai PSNR signifikan. Hal ini dikarenakan peningkatan kinerja BER pada Eb/No yang rendah (2 dB), lebih tinggi dari Eb/No yang tinggi (3 dB). Dengan demikian kesalahan diawal bit stream dapat diperbaiki sehingga kualitas citra yang direkonstruksi lebih baik (nilai PSNR yang lebih tinggi).



Gambar 7. Grafik perbandingan kinerja PSNR pengkodean gabungan dengan pengkodean tandem *code rate* ½.

5. Kesimpulan

Dari hasil simulasi, dapat diambil kesimpulan bahwa pengkodean gabungan sumber SPIHT-LDPC pada transmisi citra dapat meningkatkan kualitas citra rekonstruksi. Pada Eb/No yang rendah, peningkatan nilai BER dan PSNR sangat signifikan. Hal ini menyatakan bahwa pada kondisi kanal yang buruk, pengkodean gabungan SPIHT-LDPC lebih tahan (*less sensitive*) terhadap kesalahan kanal. Dengan demikian pemanfaatan metode pengkodean gabungan SPIHT-LDPC dapat meningkatkan daerah kerja transmisi citra. Penelitian ini selanjutnya akan dikembangkan menggunakan pengkodean sumber multi deskripsi, serta penggunaan teknik modulasi *orthogonal frequency division multiplexing* (OFDM) pada kanal frekuensi selektif.

DAFTAR PUSTAKA

[1] W. Xiang,"Joint source-channel coding for image transmission and related topic", *PhD Thesis*, Institute for Telecommunications Research Univer-sity of South Australia, Dec. 2003.

[2] A. Said and W. A. Pearlman, "A new fast and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Tech.*, vol. 6, pp. 243-250, June 1996.

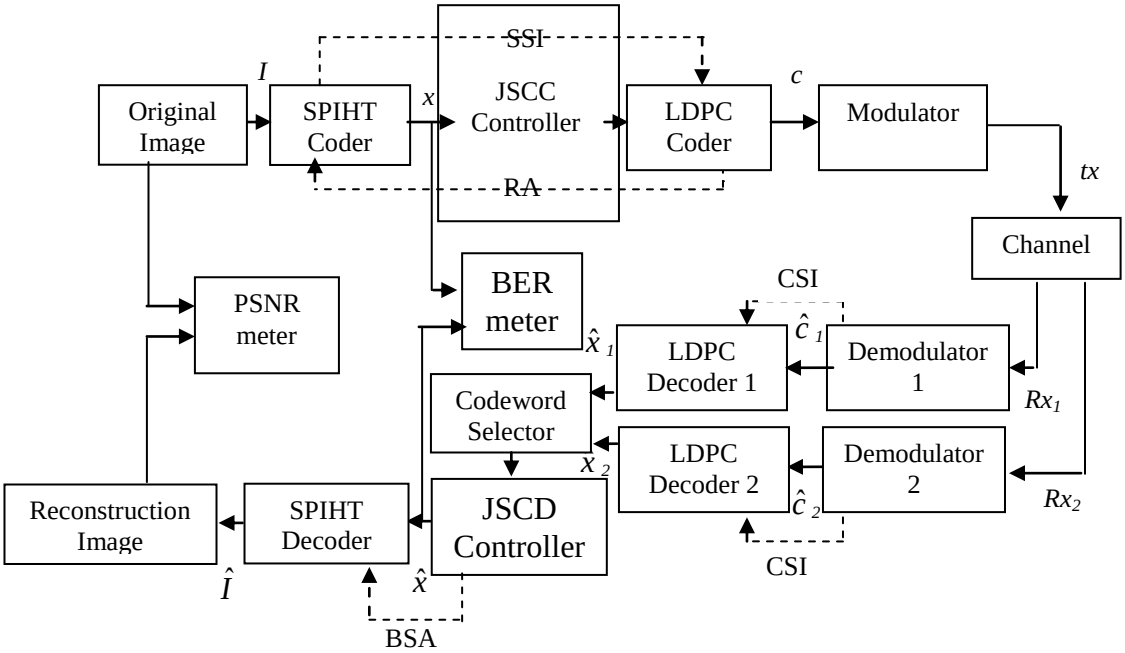
[3] R. G. Gallager, *Low-Density Parity-Check Codes*, Cambridge, MA: M.I.T. Press, 1963.

[4] I. M. Tanner, "Recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 533-547, Sep. 1981.

[5] S. Y. Chung, G. D. Forney, Jr., T. J. Richardson and R. Urbanke, "On the design of low-density parity check codes within 0.0045 dB of Shannon limit," *IEEE Comm. Lett.*, vol. 5, no. 2, pp. 58-60, Feb. 2001.

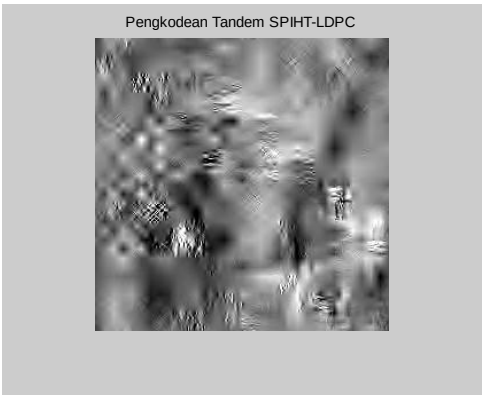
[6] T. J. Richardson and T. L. Urbanke, "The Capacity of Low-Density Parity-Check Codes Under Message-Passing Decoding," *IEEE Trans. Inform. Theory*, vol. 2, Feb. 2001.

[7] J. Richardson, M. A. Shokrollahi and R. U. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 619 – 637, Feb. 2001.



Gambar 5. Diagram sistem pengkodean gabungan SPIHT-LDPC

Hasil simulasi pengkodean gabungan SPIHT-LDPC Menggunakan teknik diversitas, pada Eb/No = 2 dB.



PSNR = 16.1951 dB

PSNR = 25.5828 dB

PSNR = 29.4597 dB

PSNR = 32.8926 dB